Інформаційні технології

УДК 004.912 **doi: 10.26906/SUNZ.2019.4.028**

О. Ю. Барковська, Д. І. Пивоварова, В. С. Сердечний, А. О. Ляшова

Харківський національний університет радіоелектроніки, Харків, Україна

ПРИСКОРЕНИЙ АЛГОРИТМ ПОШУКУ СЛІВ-ОБРАЗІВ У ТЕКСТІ З АДАПТИВНОЮ ДЕКОМПОЗИЦІЄЮ ВИХІДНИХ ДАНИХ

Алгоритми пошуку слів-образів у тексті мають широке застосування - контекстний пошук у базах та банках даних, бібліографічний пошук, пошук фрагменту тексту та його заміна при редагуванні тексту, у задачах стиску даних, алгоритмах прогнозування тощо, що зумовлює актуальність розробки нових алгоритмів, а також вдосконалення та адаптацію існуючих алгоритмів для реалізації на високопродуктивних обчислювачах. **Мета дослідження** – модифікація алгоритму Бойєра-Мура пошуку слів-образів у тексті для досягнення скорочення часу пошуку тексту завдяки використанню методів паралельних обчислень та декомпозиції вихідних даних. **Результати та висновки.** В ході роботи вдосконалено існуючий алгоритм Бойєра-Мура пошуку слів-образів у тексті завдяки використанню методів паралельних обчислень та декомпозиції вихідних даних, що призвело до скорочення часу пошуку слів-образів у текстах великого обсягу. Виконано огляд існуючих алгоритмів пошуку слів-образів у тексті, який показав найнижчу трудомісткість алгоритму Бойєра-Мура. Розроблено дві прискорені модифікації алгоритму Бойєра-Мура з простою та адаптивною декомпозицією даних. Аналіз результатів яких показав, що кількість помилкових спрацьовувань при адаптивній декомпозиції прагне до 0, на відміну від простої декомпозиції вхідних даних. Аналіз часу виконання алгоритмів показав, що на маленьких обсягах вихідного тексту, використання паралельних технологій для систем із загальною пам'яттю не є виправданим, оскільки часу на породження паралельних потоків витрачається більше, аніж на компаративні операції.

Ключові слова: слово-образ, розпаралелювання, високопродуктивна обчислювальна система, алгоритм Бойєра-Мура.

Вступ

Незважаючи на широкі можливості використання комп'ютерів для обробки різної інформації [1-8], найпопулярнішими є програми, призначені для роботи з текстом [9-16].

Програми для обробки текстової інформації діляться на кілька категорій:

- текстові редактори;
- текстові процесори;
- настільні видавничі програми;
- спеціалізовані програми обробки текстів.

Текстові редактори – це програми для створення, редагування, форматування, збереження й друку документів. Сучасний документ може містити, крім тексту, і інші об'єкти (таблиці, діаграми, малюнки тощо) (рис. 1). Прості текстові редактори призначені для створення нескладного тексту з елементами простого форматування.

Потужний текстовий редактор, що володіє більшими можливостями по обробці текстових документів (наприклад, пошук і заміна символів, засобу перевірки орфографії, вставка таблиць і ін.), зазвичай називають текстовим процесором.

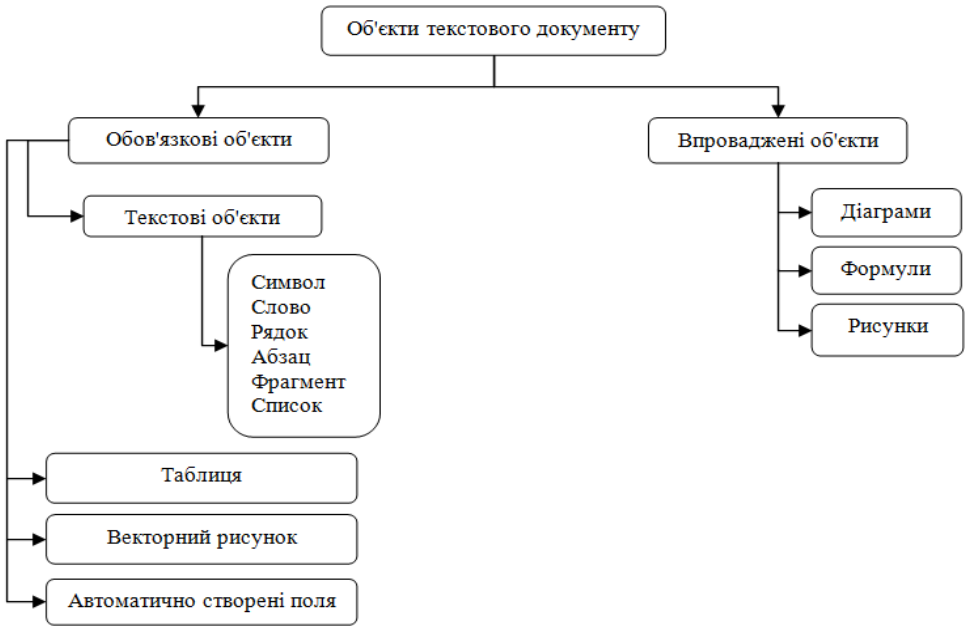


Рис. 1. Об'єкти текстових редакторів

Серед найпопулярніших форматів, із якими працюють текстові редактори, можна виділити TXT, DOC, ODT, RTF, HTML, RTF, PDF.

Основними режимами обробки текстової інформації в ТР є (рис. 2):

– уведення й редагування тексту (основний режим роботи ТР, причому редагування розуміється як будь-яка зміна в набраному тексті);

– форматування тексту (зміна зовнішнього вигляду тексту з метою створення більш ефективного й привабливого документа. Розташування рядків (довжина рядка, міжрядкова відстань, вирівнювання тексту по краю або середині й т.п.), розміри полів і сторінок – усі ці параметри встановлюються користувачем. Часто в ТР доводиться працювати з окремими фрагментами або блоками тексту. Над ними можуть бути виконані такі дії: переформатування; зміна шрифту; видалення; перенос; копіювання);

– пошук (користувач вказує ключове слово або фразу для пошуку. Результатом є виділення місця в тексті, де зустрічається дане ключове слово або фраза), заміна тексу (як правило є наступною командою після команди пошук. Результатом є заміна знайденого ключового слова або фрази при пошуку слова на інше слова або фразу, зазначену користувачем);

– орфографічний контроль;

– робота з файлами (полягає в створенні, збереженні й відкритті файла для роботи в текстовому редакторі);

– друк;

– переклад тексту;

– допомога.

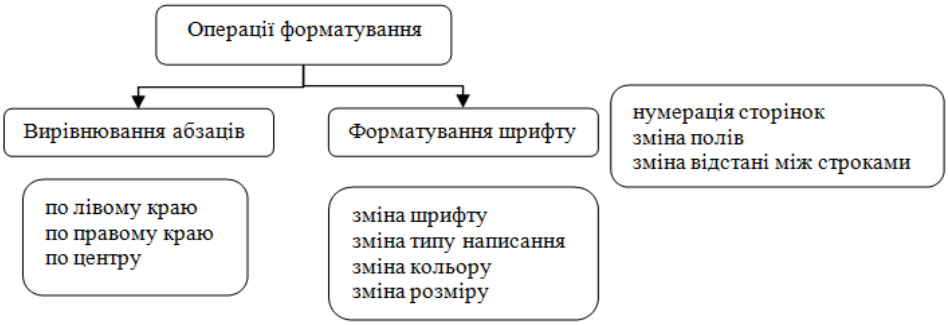


Рис. 2. Операції форматування тексту

Більшість зазначених можливостей присутні в багатьох існуючих редакторах, однак мають ряд недоліків, наприклад, швидкість роботи при пошуку й заміні ключових фраз (символів) у текстах великого розміру, а також відсутність можливості вбудованого перекладу тексту.

Постановка завдання. Метою роботи є модифікація алгоритму Бойєра-Мурадля пошуку слів-образів у тексті для досягнення прискореного пошуку тексту завдяки використанню методів паралельних обчислень та декомпозиції вихідних даних.

Досягнення поставленої мети можливе шляхом вирішення таких задач:

– огляд існуючих алгоритмів пошуку слів-образів у тексті;

– аналіз обчислювальної трудомісткості та часу реалізації розглянутих алгоритмів;

– розробка прискореної модифікації алгоритму Бойєра-Мура з простою декомпозицією даних;

– розробка прискореної модифікації алгоритму Бойєра-Мура з адаптивною декомпозицією даних;

– аналіз часу пошуку слів-образів різного розміру у текстових документах із різною вхідною кількістю символів при реалізації традиційного послідовного лінійного алгоритму, а також прискореної модифікації алгоритму БМ.

1. Аналіз існуючих методів пошуку тексту в документі

Зі збільшенням різноманіття текстових редакторів, розширенням її функціональних можливостей, зростанням обсягів інформації, яка зберігається та передається по мережі, роль задачі швидкого пошуку слова-образу у тексті стає більш актуальною. Основним застосуванням алгоритмів пошуку можуть бути контекстний пошук у базах та банках даних [15], бібліографічний пошук, пошук фрагменту тексту та його заміна при редагуванні тексту, у задачах стиску даних [16], в області захисту інформації [10, 13, 14], алгоритмах прогнозування тощо (рис. 3).

Саме внаслідок описаної важливості рішення задачі пошуку слів-образів у тексті, цією задачею займається велика кількість дослідників у різних країнах світу [9-14]. Серед існуючих алгоритмів пошуку слів-образів у текст можна виділити такі алгоритми (рис. 4):

– лінійний пошук;

– алгоритм Бойєра-Мура (БМ);

– алгоритм Кнута-Морріса;

– алгоритм Рабіна-Карпа (РК).

У роботі розглядаються такі області застосування алгоритмів пошуку, як швидкий пошук слів-образів у тексті для використання у текстових редакторах та процесорах на етапі редагування тексту (пошук тексту або пошук та заміна тексту).

Основні визначення, які використовуються в роботі, наведені нижче.

Словом-образом будемо називати послідовність символів або букв деякого алфавіту, яке необхідно знайти в тексті. Для рядку важливий склад і кількість символів в ньому, а також порядок проходження символів в рядку. Довжина рядка – це кількість символів в ньому. Наприклад, слово "ababcaba" має довжину 8. Порожній рядок – це рядок, що не містить символів, тобто має нульову довжину.

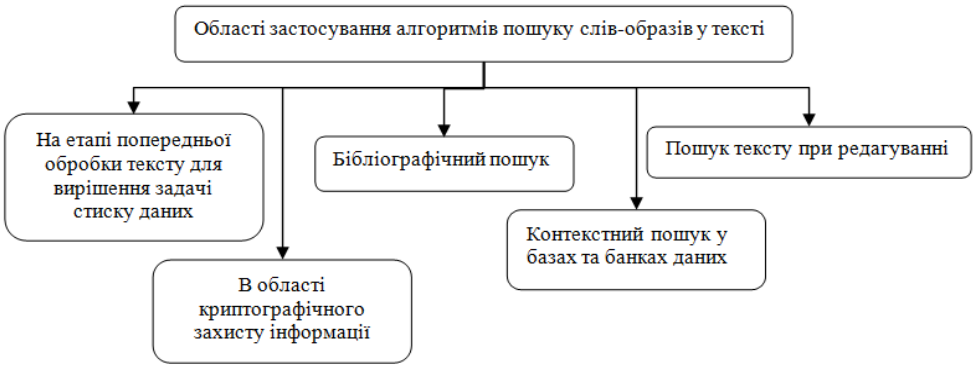


Рис. 3. Області застосування алгоритмів пошуку слів-образів у тексті

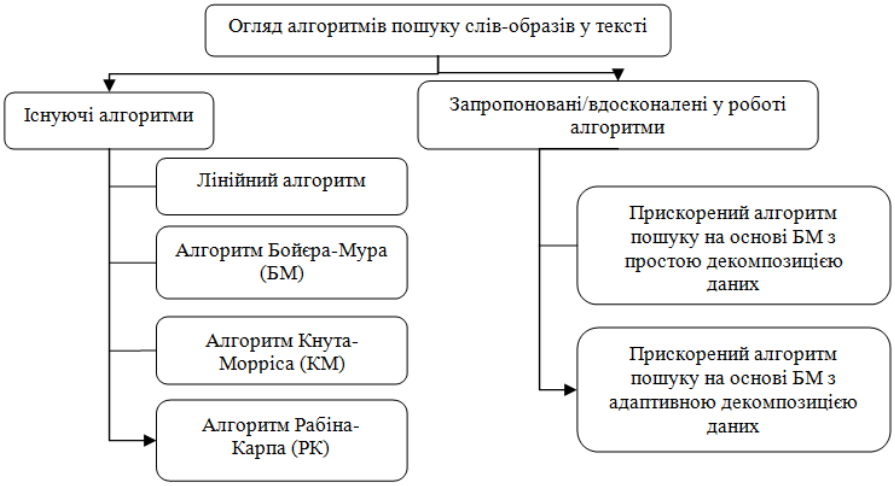


Рис. 4. Огляд алгоритмів пошуку слів-образів у тексті

Під операцією компарації розуміється порівняння символів шуканого слова і тексту для виявлення їх збігу або розходження в процесі ідентифікації рядка. Два рядки є співпадаючими, якщо вони мають один і той же склад символів, рівну довжину і однаковий порядок проходження символів. Рядки "aba" і "aab" не збігаються, тому що, хоча їх склад і довжина однакові, але порядок символів різний. Аналогічно, рядки "aaa" і "aaaaa" теж різні, оскільки мають різну довжину.

Будь-яка послідовно взята частина рядка є її підсловом. Будемо позначати символи слова буквами W з індексами, відповідними номерам цих символів в слові. При цьому символи з різними номерами можуть збігатися, тобто літери в рядку не обов'язково всі повинні бути різними. Наприклад, слово $"ababcaba"=W_1 W_2 ...W_7 W_8$, де довжина слова визначається як $j=8$, $j=\overline{1,\ell}$. Символи тексту будемо позначати буквами T з відповідними індексами: $T_1 T_2 ... T_i$, де довжина тексту визначається як $i=\overline{1,t}$. Входження заданого образу в певний текст вважається знайденим, якщо він є підсловом цього тексту і визначено його початок в тексті.

Для пошуку слова-образу в тексті за основу взято алгоритм Бойера-Мура-Хорспула, принцип роботи якого наступний: нехай заданий масив $T_з$ із t елементів і масив W з ℓ елементів, причому $0 \le \ell < t$. Пошук рядка виявляє входження W в T. Обидва масиви містять символи, так що $T_з$ можна вважати деяким текстом або рядком, а W – образом, який необхідно знайти.

Етапи роботи алгоритму [9]:

– формування таблиці d, яка буде використовуватися для зміщення образу по рядку;

– пошук образу в рядку.

Порівняння символів починається з кінця образу, а не з початку. Ефективність алгоритму Бойри-Мура-Хорспула обумовлена тим, що вдається пропускати ті частини тексту, які свідомо не беруть участь в успішному зіставленні.

Трудомісткість алгоритмів пошуку слів-образів у тексті визначається кількістю компарації символів у шуканому слові-образі довжиною ℓ із вихідним текстом довжиною t, яка має бути виконана у процесі роботи алгоритму.

Середня трудомісткість розглянутих існуючих алгоритмів наведена у табл. 1.

Таблиця 1 – **Середня трудомісткість існуючих алгоритмів пошуку слів-образів у тексті**

Алгоритм пошуку слів-образів у тексті	**Середня трудомісткість алгоритму**
Лінійний пошук	$O((t-\ell)\times\ell)$
Алгоритм Бойєра-Мура (БМ)	$O(t/\ell))$
Алгоритм Кнута-Морріса	$O(t+\ell)$
Алгоритм Рабіна-Карпа (РК)	$O((t-\ell)\times\ell)$

Оскільки найменшу трудомісткість має алгоритм Бойєра-Мура, його принцип роботи буде взято за основу при подальшому вдосконаленні для досягнення меншого часу роботи наведеного методу. Існуючі модифікації алгоритму БМ наведені у [9, 11, 12].

2. Рішення поставленої задачі

2.1. Розробка прискореного алгоритму на основі БМ з простою декомпозицією вихідних даних. Аналіз алгоритму БМ показує можливість паралельної реалізації пошуку слів-образів на багатопроцесорній обчислювальній системі із загальною пам'яттю. Такі системи, як правило, складаються з декількох однорідних процесорів і масиву загальної пам'яті. Кожен з процесорів має прямий доступ до будь-якої комірки пам'яті, причому швидкість доступу до пам'яті для всіх процесорів однакова. Зазвичай процесори підключаються до пам'яті за допомогою загальної шини або за допомогою спеціальних комутаторів. У багатопроцесорної системі із загальною пам'яттю один процесор здійснює запис в конкретну комірку пам'яті, а інший процесор робить зчитування з цього місця пам'яті. Щоб забезпечити узгодженість даних і синхронізацію процесів, обмін часто реалізується за принципом взаємно виключного доступу до спільної пам'яті. Далі в областях розпаралелювання породжується ряд паралельних потоків, які виконуються на різних процесорах, які входять в обчислювальну систему [1-8].

Робота починається з ініціалізації і роботи головного потоку (процесу), який у міру необхідності створює і виконує паралельні потоки, передаючи їм необхідні дані. Паралельні потоки з однієї паралельної області програми виконуються незалежно один від одного. Після завершення виконання паралельних потоків, управління програмою знову передається головному потоку. Для прискорення роботи розглянутих методів пошуку слів-образів у тексті, пропонується їх виконання в паралельному вигляді. За основу взято алгоритм БМ, оскільки саме він має найменшу трудомісткість. Для можливості ефективного розпаралелювання виконується декомпозиція вхідних даних, оскільки залежності між даними немає. Тобто, кожна підгрупа даних після декомпозиції може бути оброблена незалежно одна від одної і залежності від результатів операцій, що знаходяться в попередніх групах, немає. Важливою умовою є отримання вірного результату, тобто знаходження 100% шуканих слів-образів у тексті.

Згадаємо, що алгоритм БМ має два етапи. При створенні паралельної модифікації алгоритму пропонується перший етап – створення таблиці зміщення, виконувати на головному процесорі P_0, а вже починаючи з другого етапу – пошук слова в тексті виконувати незалежно на різних обчислювальних процесорах P_i, де $i = \overline{0,n}$, на різних наборах даних.

Для рівномірного обчислювального навантаження на потоки, обсяг обчислень для кожного з них повинен бути приблизно однаковий. Ґрунтуючись на цьому твердженні, можна зробити висновок, що розпаралелювання запропонованого методу пошуку є обґрунтованим, так як робота другого етапу алгоритму БМ буде однаковою для кожного обчислювача і буде залежати лише від розміру підзадач, обробка яких виконується паралельно. Робота кожного процесора буде виконуватись на даних однакового розміру, буде мати однакову обчислювальну складність що дозволяє мінімізувати наявність інформаційних зв'язків між підзадачами та вирівняти обчислювальне навантаження для кожного процесора.

Нехай задано вхідний текст *T,* де одним із символів є T_i, $i = \overline{(1,t)}$. Тоді шукане слово-образ *W,* яке складається із символів W_j, буде включати *j* символів: $j = \overline{1,\ell}$. На початку роботи алгоритму виконується завантаження тексту та слова-образу у загальну пам'ять центрального процесора. На головному процесорі починається послідовна частина роботи алгоритму, а саме – формування таблиці зміщення (рис. 5). Після формування таблиці, починається породження паралельних потоків $P = [P_0, P_1, \ldots, P_c]$, $c = \overline{1,n-1}$ та ініціюється звернення усіх процесорів до загальної пам'яті для отримання своєї інформації, а саме – таблиці зміщення та блоку тексту, розмір якого визначається за правилом: потік P_c отримує набір даних в інтервалі $\left[T \cdot t/(n/c)+1, \ldots, T \cdot t/(n/c+1)\right]$.

Після отримання різних наборів даних однакового розміру та єдиної для всіх потоків інструкції, починається обчислювальний другий етап алгоритму БМ. Після завершення обробки тексту усіма потоками починається етап об'єднання результатів та знайдених слів-образів. Характерною рисою для цього підходу є жорстке закріплення блоків даних за кожним потоком, не зважаючи на розмір шуканого слова-образу W_i.

2.2. Розробка прискореного алгоритму на основі БМ з адаптивною декомпозицією вихідних даних. Ще однією прискореною модифікацією алгоритму БМ є прискорена модифікація алгоритму БМ з адаптивною декомпозицією даних, що свідчить про те, що при визначенні блоку даних для кожного потоку враховується розмір слова-образу.

Виконання алгоритму майже ідентичне попередньому підходу, окрім правила формування блоків даних перед тим, як кожен потік перейде до виконання другого компаративного етапу алгоритму БМ. Розмір блоку тексту для кожного потоку визначається за правилом: потік P_c отримує набір даних в інтервалі $\left[T \cdot t/(n/c)+1, \ldots, T \cdot t/(n/c+1)\right]$.

Тобто, розмір блоку даних для кожного потоку збільшився на ℓ. Після завершення компаративного етапу кожним потоком паралельної області, управління передається основному потоку, який займається збором частин обробленого тексту. Після цього виконання знову переходить в паралельну область. Механізм розпаралелювання для обчислювальної системи із загальною пам'яттю, породження паралельних потоків, декомпозиція вихідних даних у відповідності із правилом fork-join [3, 4] зображено на рис. 5.

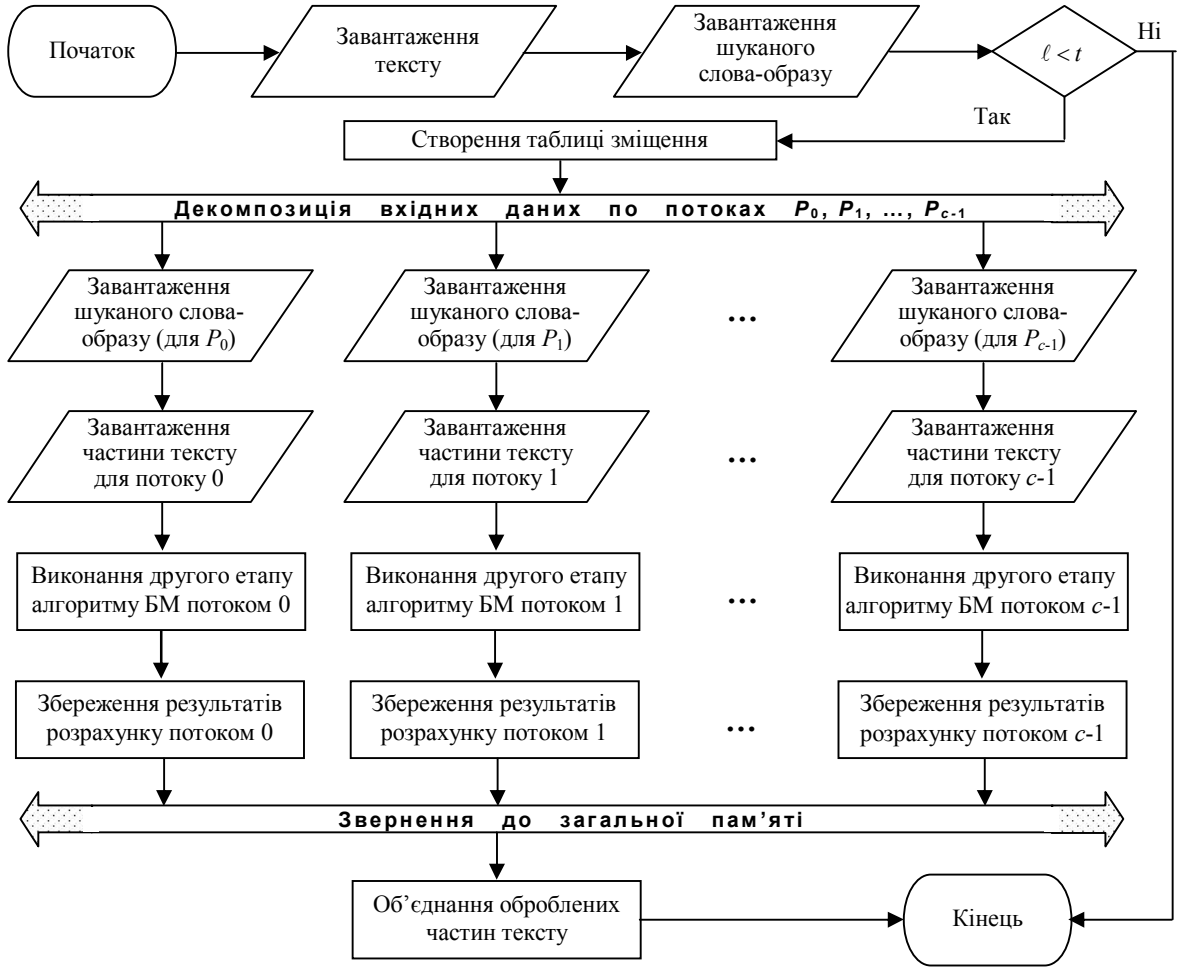


Рис. 5. Прискорена модифікація алгоритму БМ з адаптивною декомпозицією даних

3. Аналіз отриманих результатів

Тестування функції пошуку та пошуку і заміни елементів було проведено на різних тестових наборах: послідовний пошук одного символу у тексті розміром 1902 символи; послідовний пошук чотирьох символів у тексті розміром 1902 символи; послідовний пошук дев'яти символів у тексті розміром 1902 символи; послідовний пошук одного символу у тексті розміром 18598 символи; послідовний пошук чотирьох символів у тексті розміром 18598 символи; послідовний пошук дев'яти символів у тексті розміром 18598 символи. В результаті виконання пошуку елементів були отримані результати для тексту розміром 1 сторінка (1902 символів) (рис. 6). Аналіз отриманих результатів показав, що при збільшенні шуканих символі, час пошуку збільшується так само, як і збільшується час пошуку при збільшенні обсягу вихідного тексту.

Кількість помилкових спрацьовувань та час виконання пошуку слів-образів, а також пошуку та заміни слів-образів у тексті із використанням технології паралельного програмування на системах із загальною пам'яттю при різних запропонованих підходах до декомпозиції даних, наведено у таблицях 2-4, де позначено: $K_{ПОЧ}$ – кількість символів в початковому тексті; $K_{ПОШ}$ – кількість символів для пошуку; $T_{ВИТР}$ – витрачений час (мс); $K_{ПОМ}$ – кількість помилкових спрацьовувань. Отримані дані доводять відсутність необхідності використання більше одного обчислювача при роботі з невеликими обсягами даних. Адаптивна декомпозиція даних завдяки тому, що при розподіленні вихідного тексту по обчислювачам враховується довжина шуканого слова, забезпечує швидкий та безпомилковий пошук слів-образів у тексті.

Таблиця 2 – **Витрачений час послідовного пошуку слів-образів у тексті**

$K_{ПОЧ}$	$K_{ПОШ}$	$T_{ВИТР}$
1902	1 символ	0,29
1902	1 слово (4 символи)	0,30
1902	1 слово (9 символи)	0,98
18598	1 символ	5
18598	1 слово (4 символи)	8,9
18598	1 слово (9 символи)	11,6

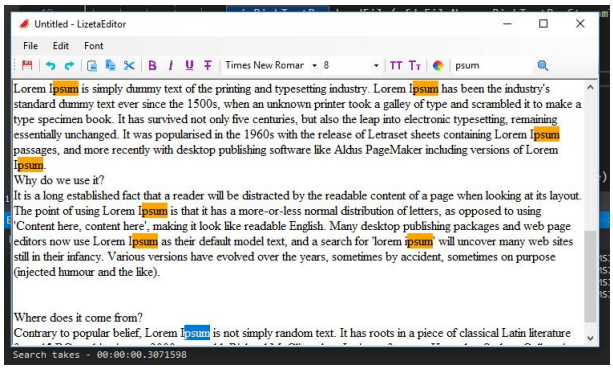
Таблиця 3 – **Витрачений час прискореного пошуку слів-образів у тексті при простій декомпозиції даних**

$K_{ПОЧ}$	$K_{ПОШ}$	$T_{ВИТР}$	$K_{ПОМ}$
1902	1 символ	0,32	0
1902	1 слово (4 символи)	0,31	2
1902	1 слово (9 символи)	0,87	4
18598	1 символ	4,5	0
18598	1 слово (4 символи)	4,5	3
18598	1 слово (9 символи)	4	4

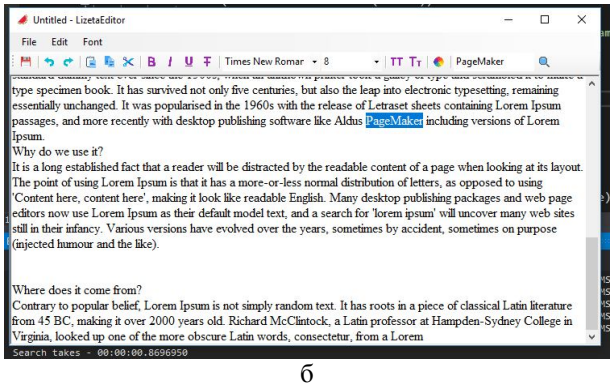
Таблиця 4 – **Витрачений час прискореного пошуку слів-образів у тексті при адаптивній декомпозиції даних**

$K_{поч}$	$K_{пош}$	$T_{витр}$	$K_{пом}$
1902	1 символ	0,31	0
1902	1 слово (4 символи)	0,31	0
1902	1 слово (9 символи)	0,85	0
18598	1 символ	5	0
18598	1 слово (4 символи)	4,65	0
18598	1 слово (9 символи)	3,56	0

Графік отриманого прискорення у порівнянні із послідовною реалізацією наведено на рис. 6.



а



б

Рис. 6. Пошук слів-образів у тексті розміром 1902 символи:
а – розмір слова-образу – 1 символ,
б – розмір слова-образу – 9 символів

На підставі отриманих таблиць 2-4, було побудовано діаграми для аналізу результатів, що зображені на рис. 7, 8. Таким чином, було виявлено, що зі збільшенням розміру слова для пошуку прямо пропорційно зменшувався час на його пошук по усьому тексту через задіяння більшої кількості ядер.

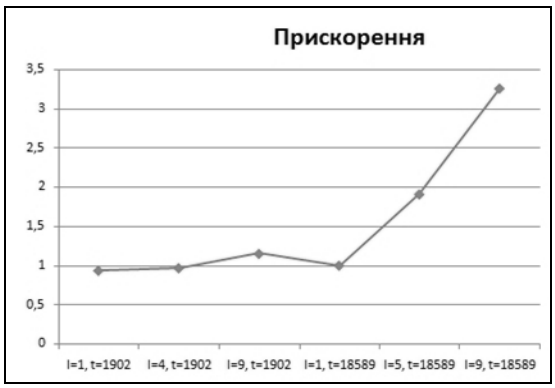


Рис. 7. Графік отриманого прискорення для методу адаптивної декомпозиції відносно послідовної реалізації алгоритму

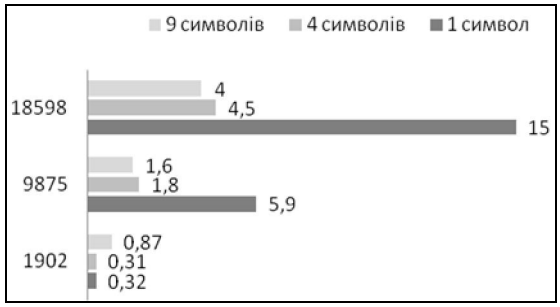


Рис. 8. Діаграма результатів пошуку слів-образів у тексті

Висновки

В ході роботи було вдосконалено існуючий алгоритм Бойєра-Мура пошуку слів-образів у тексті для скорочення часу пошуку завдяки використанню методів паралельних обчислень та декомпозиції вихідних даних. Було виконано огляд існуючих алгоритмів пошуку слів-образів у тексті, який показав найнижчу трудомісткість алгоритму БМ. Було розроблено дві прискорені модифікації алгоритму Бойєра-Мура з простою декомпозицією даних та адаптивною декомпозицією даний, аналіз результатів яких показав, що кількість помилкових спрацьовувань при адаптивній декомпозиції прагне до 0, на відміну від простої декомпозиції вхідних даних. Аналіз часу виконання алгоритмів показав, що на маленьких обсягах вихідного тексту, використання паралельних технологій для систем із загальною пам'яттю не є виправданим, оскільки часу на породження паралельних потоків витрачається більше, аніж на компаративні операції.

СПИСОК ЛІТЕРАТУРИ

1. Левин М.П. Параллельное программирование с использованием OpenMP / М.П. Левин. – БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий, 2008. – 120с.
2. Антонов А.С. Параллельное программирование с использованием технологии OpenMP. – М.: МГУ, 2009. – 77 с.
3. Эндрюс Г. Р. Основы многопоточного, параллельного и распределенного программирования / Г.Р. Эндрюс. – М.: Издательский дом "Вильямс", 2003. – 512 с.
4. Миллер Р. Последовательные и параллельные алгоритмы: Общий подход / Р. Миллер, Л. Боксер. – БИНОМ. Лаборатория знаний, 2006. – 406 с.
5. V. Kharchenko, A. Kovalenko, O. Siora, V. Sklyar, "Security assessment of FPGA-based safety-critical systems: US NRC requirements context", *Proceedings of IEEE Information and Digital Technologies Conference (IDT)*, pp. 132-138, July 2015. DOI: http://doi.org/10.1109/DT.2015.7222963
6. Kuchuk G., Kovalenko A., Komari I.E., Svyrydov A., Kharchenko V. Improving big data centers energy efficiency: Traffic based model and method. Studies in Systems, Decision and Control, vol 171. Kharchenko, V., Kondratenko, Y., Kacprzyk, J. (Eds.). Springer Nature Switzerland AG, 2019. Pp. 161-183. DOI: http://doi.org/10.1007/978-3-030-00253-4_8

7. Kharchenko, V., Illiashenko, O., Kovalenko, A., Sklyar, V., Boyarchuk, A. Security informed safety assessment of NPP I&C systems: GAP-IMECA technique. Proc. 22th Int. Conf. on Nuclear Engineering (ICONE), Prague, Republic, 7-11 Jul. 2014, pp. 1-9. doi: http://doi.org/10.1115/ICONE22-31175.
8. Кучук Г.А. Метод мінімізації середньої затримки пакетів у віртуальних з'єднаннях мережі підтримки хмарного сервісу / Г.А. Кучук, А.А. Коваленко, Н.В. Лукова-Чуйко // Системи управління, навігації та зв'язку. – Полтава . ПНТУ, 2017. – Вип. 2(42). – С. 117-120.
9. Z. Xiong, "A Composite Boyer-Moore Algorithm for the String Matching Problem," 2010 International Conferenceon Parallel and Distributed Computing, Applicationsand Technologies, Wuhan, 2010, pp. 492-496.doi: 10.1109/PDCAT.2010.58
10. A. Domínguez, P. P. Carballoand A. Núñez, "Programmable SoC platform for deep packet inspection using enhanced Boyer-Moore algorithm," 2017 12th International Symposiumon Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), Madrid, 2017, pp. 1-8. doi: 10.1109/ReCoSoC.2017.8016159
11. Y. Jeong, M. Lee, D. Nam, J. Kimand S. Hwang, "High Performance Parallelization of Boyer-Moore Algorithmon Many-Core Accelerators," 2014 International Conference on Cloud and Autonomic Computing, London, 2014, pp. 265-272. doi: 10.1109/ICCAC.2014.20
12. X. Zhaand S. Sahni, "GPU-to-GPU and Host-to-Host Multipattern String Matching on a GPU," in IEEE Transactionson Computers, vol. 62, no. 6, pp. 1156-1169, June 2013. doi: 10.1109/TC.2012.61
13. V. Gupta, M. Singhand V. K. Bhalla, "Pattern matching algorithms for intrusion detection and prevention system: A comparative analysis," 2014 International Conferenceon Advancesin Computing, Communications and Informatics (ICACCI), NewDelhi, 2014, pp. 50-54. doi: 10.1109/ICACCI.2014.6968595
14. Y. Wangand H. Kobayashi, "AnImproved Technology for Content Matching Intrusion Detection System," 2006 Int. Conf. on Software in Telecommunications and Computer Networks, Split, 2006, pp. 238-241. doi: 10.1109/SOFTCOM.2006.329755
15. V. R. Krishna and P. V. Kumar, "Optimal pattern search for sequence databases," 2010 3rd Int. Conf. on Advanced Computer Theory and Engineering (ICACTE), Chengdu, 2010, pp. V2-654-V2-658. doi: 10.1109/ICACTE.2010.5579518
16. P. Lin, S. Liu, L. Zhangand P. Huang, "Compressed Patter nMatchingin DNA Sequences Using Multithreaded Technology," 2009 3rd International Conference on Bioinformatics and Biomedical Engineering, Beijing, 2009, pp. 1-4. doi: 10.1109/ICBBE.2009.5162550

Рецензент: д-р техн. наук, проф. С. О. Ляшенко,
Харківський національний технічний університет сільського господарства ім. Петра Василенка, Харків
Received (Надійшла) 18.06.2019
Accepted for publication (Прийнята до друку) 21.08.2019

Ускоренный алгоритм поиска слов-образов в тексте с адаптивной декомпозиций выходных данных

О. Ю. Барковская, Д. И. Пивоварова, В. С. Сердечный, А. А. Ляшова

Алгоритмы поиска слов-образов в тексте имеют широкое применение - контекстный поиск в базах и банках данных, библиографический поиск, поиск фрагмента текста и его замена при редактировании, в задачах сжатия данных, алгоритмах прогнозирования и т.п., что обуславливает актуальность разработки новых алгоритмов, а также совершенствования и адаптации существующих алгоритмов для реализации на высокопроизводительных вычислителях. **Цель исследования** - модификация алгоритма Бойера-Мура для поиска слов-образов в тексте для достижения сокращения времени поиска текста благодаря использованию методов параллельных вычислений и декомпозиции исходных данных. **Результаты и выводы.** В ходе работы усовершенствован существующий алгоритм Бойєра-Мура поиска слов-образов в тексте благодаря использованию методов параллельных вычислений и декомпозиции исходных данных. Выполнен обзор существующих алгоритмов поиска слов-образов в тексте, который показал самую низкую трудоемкость алгоритма Бойєра-Мура. Разработаны две ускоренные модификации алгоритма Бойєра-Мура с простой и адаптивной декомпозицией данных.Анализ результатов показал, что количество ложных срабатываний при адаптивной декомпозиции стремится к 0, в отличие от простой декомпозиции входных данных. Анализ времени выполнения алгоритмов показал, что на маленьких объемах исходного текста, использование параллельных технологий для систем с общей памятью не является оправданным, поскольку времени на порождение параллельных потоков тратится больше, чем на компаративные операции.

Ключевые слова: слово-образ, распараллеливание, высокопроизводительная вычислительная система, алгоритм Бойєра-Мура.

Accelerated algorithm for word search in text with adaptive decomposition of the output

O. Yu. Barkovska, D. I. Pyvovarova, V. S. Serdechnyi, A. A. Liashova

Word search algorithms in text are widely used - contextual search in databases and databases, bibliographic search, search for a text fragment and its replacement when editing, in data compression tasks, prediction algorithms, etc., which determines the urgency of developing new algorithms , as well as improving and adapting existing algorithms for implementation on high-performance computers. **The purpose of the research** is to modify the Boyer-Moore algorithm to search for word-images in text to achieve reduced text search time by using parallel computation methods and decomposing raw data. **Results and conclusions.** In the course of the work, the existing Boyer-Moore algorithm for word search in text has been improved by the use of parallel computation methods and decomposition of raw data. An overview of the existing word search algorithms in the text was performed, which showed the lowest complexity of the Boyer-Moore algorithm. Two accelerated modifications of the Boyer-Moore algorithm with simple and adaptive data decomposition are developed. Analysis of the results showed that the number of false positives in adaptive decomposition tends to 0, as opposed to simple decomposition of the input data. An analysis of the execution time of the algorithms showed that on small volumes of the source text, the use of parallel technologies for systems with shared memory is not justified, since the time for generation of parallel flows is spent more than for comparative operations.

Keywords: word-image, parallelization, high-performance computing system, Boyer-Moore algorithm.